Ben-Gurion University of the Negev
The Faculty of Natural Sciences
The Department of Computer Science

# Improving Koopman-Based Sequential Multifactor Disentanglement with Single Static Mode and Latent Space Refinement

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Science

Amos Haviv Hason

Under the supervision of:
Senior Lecturer Dr. Omri Azencot

December 2024

Ben-Gurion University of the Negev
The Faculty of Natural Sciences
The Department of Computer Science

December 2024

# Improving Koopman-Based Sequential Multifactor Disentanglement with Single Static Mode and Latent Space Refinement

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

Master of Science

Amos Haviv Hason

Under the supervision of:
Senior Lecturer Dr. Omri Azencot

Amos Haviv Hason
Author

Date

Dr. Omri Azencot
Supervisor

Date

Prof. Eyal Shimony
Departmental graduate students committee chair

Date

# IMPROVING KOOPMAN-BASED SEQUENTIAL MULTIFACTOR DISENTANGLEMENT WITH SINGLE STATIC MODE AND LATENT SPACE REFINEMENT

**Amos Haviv Hason**

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

AT

BEN-GURION UNIVERSITY OF THE NEGEV

2024

## ABSTRACT

Representation learning aims to automatically discover useful features from raw data to enhance tasks such as classification and prediction. Disentangled representations, where distinct features correspond to independent factors of variation in the data, offer a promising approach to improving model generalization, especially in sequential data. However, learning such representations in an unsupervised manner remains challenging, particularly due to the scarcity of labeled data. In this work, we focus on the Structured Koopman Disentanglement (SKD) model, which applies dynamic mode decomposition (DMD) using Koopman theory within an autoencoder to achieve unsupervised sequential disentanglement. We identify key issues in the original SKD implementation and propose a reimplementation that resolves these shortcomings. Furthermore, we introduce the Single Static Mode Structured Koopman Disentanglement (SSM-SKD) model, which incorporates modifications to the DMD and latent space extraction stages. To enhance exploration of the latent space, we also propose a greedy latent space exploration algorithm. Our model is evaluated on four sequential multifactor disentanglement datasets, demonstrating its effectiveness compared to the original SKD model. Additionally, we contribute to solving the ongoing reproducibility challenge in machine learning by proposing a comprehensive reporting standard for experimental environments.

## Preface

This work has been performed partly under fire amid the ongoing Swords of Iron war, an existential war that has shaken Israel. The tragic events of October 7, 2023 and the events which followed that day have affected every person in Israel, including myself. Throughout my research, my heart has always been with the victims of the horrific terrorist and rocket attacks and their families, our fallen protectors and their families, our hostages which are yet to be freed and their families, and our displaced citizens that are yet unable to return to their homes.

In the beginning of the war I volunteered to develop and compose a musical theme for a free video game for Israeli kids. I also volunteered in a civilian project which aimed to develop a deep learning-based solution for the detection of rogue aerial vehicles. We aimed to provide this solution for the military, but unfortunately we did not manage to get it to work well in the field.

By performing this research I achieved a lot of knowledge on how research in machine learning is performed. A lot of times I felt stuck, but by being stuck I was able to realize stuff that I would probably not have been able to realize otherwise.

My original research idea for my thesis was to adjust the Structured Koopman Disentanglement (SKD) model to the musical task of timbre transfer by learning a representation that disentangles pitch and timbre. Despite three months of research, I reached a dead-end. I got frustrated and was desperately in need of a shift, and this is when my supervisor, Dr. Omri Azencot, suggested the idea that led to this thesis.

In the last three weeks of my experimental work, as part of this research, I had the honor to work on a potential publication, which I hope will turn actual.

I would like to thank my family, Dr. Azencot and his research group members, my graduate school colleagues, the administrative and technical staffs of Ben-Gurion University of the Negev, and the general scientific community for guiding and assisting me in my master's program and research.

# CONTENTS

## List of Figures

## List of Tables

# 1 Introduction

## 1.1 Representation Learning

**Representation learning** [3] is a subfield of machine learning that focuses on automatic discovery of the most useful features or representations from raw data to improve the performance of downstream tasks, such as classification, prediction, or decision-making. Instead of relying on manually designed features, representation learning methods aim to learn these features directly from the data in a meaningful and often lower-dimensional form, which is referred to as the learned latent space.

## 1.2 Sequential Disentanglement

A **disentangled representation** is a type of data representation where the learned features correspond to distinct, interpretable factors of variation in the data. In other words, each dimension or component of the representation captures a specific, independent aspect of the underlying structure of the data, such as shape, color, or position in images, or separate time-invariant (static) and time-varying (dynamic) factors in sequential data. Such a representation is desirable since it may aid in achieving better generalization in downstream tasks [3].

A challenge in disentanglement learning is the poor availability of labeled data, especially in real-world settings. Hence, it is of interest to tackle disentanglement learning in an unsupervised manner. One promising approach to sequential disentanglement that was recently proposed, in the unsupervised Structured Koopman Disentanglement (SKD) model [4], is to perform dynamic mode decomposition (DMD) [36; 35; 46] through the application of Koopman theory [19; 7] in the bottleneck of an autoencoder network [9].

## 1.3 Our Contributions

Our contributions in this work are as follows:

1. We point out key issues in the original implementation of SKD and suggest a reimplementation that addresses these.
2. We propose the Single Static Mode Structured Koopman Disentanglement (SSM-SKD) model, which is based on our reimplementation of SKD along with modifications to the DMD and latent space extraction stages of the model.
3. We propose a greedy latent space exploration algorithm to be used in tandem with SSM-SKD.
4. We evaluate SSM-SKD on four sequential multifactor disentanglement datasets and compare it to SKD's results on one of them.
5. We address an aspect of the reproducibility problem in machine learning research by suggesting a new standard for comprehensive reporting of environment details.

# 2 Background

## 2.1 Koopman Theory and Dynamic Mode Decomposition (DMD)

**Koopman theory** provides a theoretical framework for representing nonlinear dynamical systems using a linear operator (the Koopman operator) in an infinite-dimensional space. This operator acts on observable functions of the system, mapping them forward in time. The Koopman operator describes how measurements (or functions of state) evolve over time. It transforms the state space of a nonlinear system into a linear framework, which allows the application of linear analysis techniques to inherently nonlinear systems. The key feature is that it does not linearize the system itself, but rather provides a linear representation of its dynamics through observables. Koopman theory is mostly concerned with the spectral properties of this operator, including its eigendecomposition, which provides insight into the long-term behavior of the system.

Given a discrete-time system:

$$\mathbf{x}_{n+1} = \mathbf{F}(\mathbf{x}_n), \quad \mathbf{x} \in \mathbb{R}^d,$$

with $\mathbf{F} : \mathbb{R}^d \to \mathbb{R}^d$, let $g : \mathbb{R}^d \to \mathbb{C}$ be an observable.

The Koopman operator $\mathcal{K}$ acts on $g$ as:

$$\mathcal{K}g(\mathbf{x}) = g(\mathbf{F}(\mathbf{x})).$$

Eigenfunctions $\phi(\mathbf{x})$ satisfy:

$$\mathcal{K}\phi(\mathbf{x}) = \lambda\phi(\mathbf{x}),$$

with $\lambda$ as eigenvalues. Any observable $g(\mathbf{x})$ can be expanded as:

$$g(\mathbf{x}) = \sum_k c_k \phi_k(\mathbf{x}),$$

yielding dynamics:

$$\mathcal{K}^n g(\mathbf{x}) = \sum_k c_k \lambda_k^n \phi_k(\mathbf{x}).$$

**DMD** is a numerical algorithm that approximates the Koopman operator from data. It was initially developed in fluid mechanics for extracting dynamic modes from time-resolved data, such as experimental flow measurements or numerical simulations. DMD approximates the finite-dimensional representation of the Koopman operator by analyzing snapshots of the system over time. It identifies dominant modes (dynamic modes) and their corresponding eigenvalues, which describe how the system evolves. In essence, DMD computes an eigendecomposition of a linear operator that approximates the behavior of the system, making it particularly useful for studying high-dimensional time-dependent data. DMD is often used in practical scenarios to decompose signal data into modes associated with particular frequencies and growth/decay rates, akin to Fourier decomposition, but for dynamic systems.

Koopman theory is a theoretical framework focused on the operator that describes the evolution of observables in an infinite-dimensional space, while DMD is a data-driven algorithm designed to approximate this operator in practice using finite data. Koopman theory deals with the underlying dynamics of nonlinear systems and their linear representations, whereas DMD focuses on extracting dominant modes from time-series data and approximating the dynamics. DMD is often viewed as a practical application of Koopman theory but in finite dimensions and based on observable data, making it useful in fields like fluid dynamics, video processing, and system identification.

### 2.2 Structured Koopman Disentanglement (SKD)

#### 2.2.1 Description

**SKD** is an autoencoder model with a Koopman module in its bottleneck. The encoder and decoder architectures are not particular and may freely be chosen according to the domain of the data being modeled. In the scope of this work the encoder consists of a convolutional neural network (CNN) followed by a long short-term memory (LSTM) network, while the decoder consists of an LSTM network followed by a CNN.

Consider an input tensor $X$ of shape $[N, T, *]$, where $N$ is the batch size, $T$ is the sequence length, and $*$ denotes the element dimensions. In the scope of this work, dealing with video frames, the shape is particularly $[N, T, C, H, W]$, where $C$ is the number of color channels, $H$ is the frame's height, and $W$ is the frame's width. Passing $X$ through the encoder results in a latent tensor $Z$ of shape $[N, T, K]$, where $K$ serves as the dimension of Koopman operator, which is a square matrix of size $K \times K$.

In order to approximate the Koopman operator, we take the first $T-1$ and last $T-1$ frames of $Z$, denoted by $Z_{0:T-2}$ and $Z_{1:T-1}$, respectively, then we reshape both tensors to matrices of size $[N \times (T-1), K]$, denoted by $[\![Z]\!]_{0:T-2}$ and $[\![Z]\!]_{1:T-1}$, respectively, and compute a solution to the least squares problem for the linear system $[\![Z]\!]_{0:T-2}\mathcal{K} = [\![Z]\!]_{1:T-1}$. Hence, the resulting matrix $\mathcal{K}$ is the approximated Koopman operator of the batch.

We then use $\mathcal{K}$ to get the latent prediction tensor through batch matrix product by broadcasting $\mathcal{K}$: $Z_{1:T-1_{\text{pred}}} = Z_{0:T-2}\mathcal{K}$. We then concatenate the first frame to the latent prediction tensor: $Z_{\text{pred}} = Z_0 + \!\!\!+ \, Z_{1:T-1_{\text{pred}}}$. On $Z$ we apply dropout: $Z_{\text{do}} = \text{Dropout}_p(Z)$, where $p$ is the dropout probability. Finally, we separately pass $Z_{\text{pred}}$ and $Z_{\text{do}}$ through the decoder to get the ambient prediction tensor $X_{\text{pred}}$ and reconstructed tensor $X_{\text{rec}}$, respectively.

We use four loss terms for the backward pass - three of them are mean squared error (MSE) loss terms, and one of them is a Koopman spectral loss term:

1. The latent prediction loss term:

$$\mathcal{L}_{\text{pred}_Z} = \text{MSE}(Z_{\text{pred}}, Z) \tag{1}$$

2. The ambient prediction loss term:

$$\mathcal{L}_{\text{pred}_X} = \text{MSE}(X_{\text{pred}}, X) \tag{2}$$

3. The reconstruction loss term:

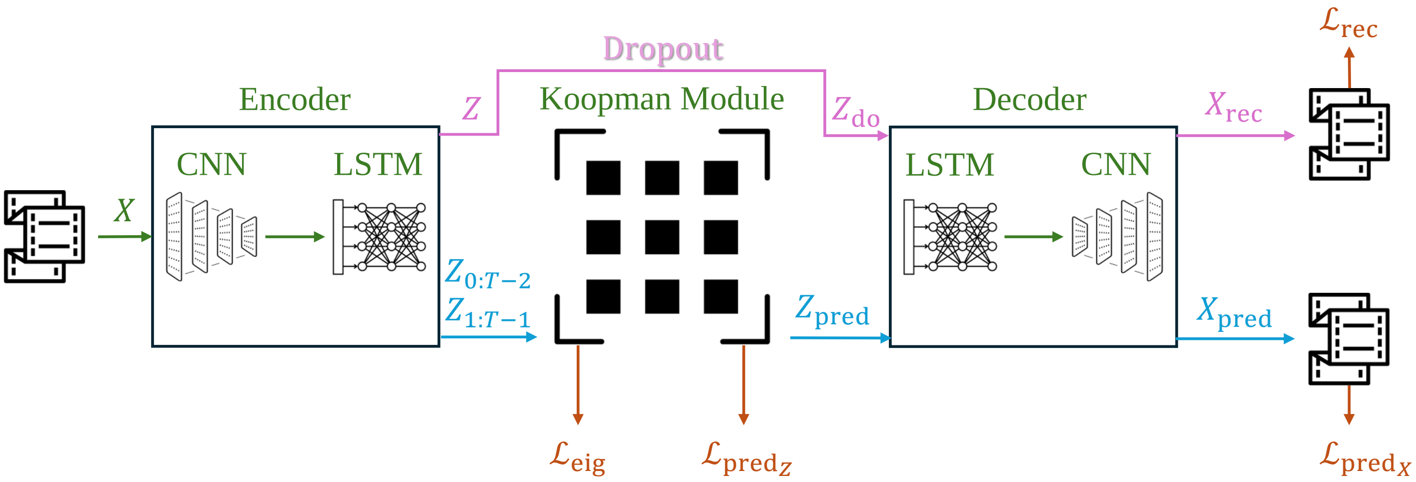
$$\mathcal{L}_{\text{rec}} = \text{MSE}(X_{\text{rec}}, X) \tag{3}$$

4. The Koopman spectral loss term:

$$\mathcal{L}_{\text{eig}} = \frac{1}{|S|}\sum_{\lambda \in S} |\lambda - 1|^2 + \frac{1}{|D|}\sum_{\lambda \in D} \begin{cases} \text{Re}(\lambda), & \text{if } \text{Re}(\lambda) > \alpha \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

   Here, following an eigendecomposition of $\mathcal{K}$, $S$ is subset of the $s$ (a user-defined hyperparameter) eigenvalues of $\mathcal{K}$ which are closest to 1 by Euclidean distance, $D$ is subset of the rest of eigenvalues of $\mathcal{K}$, and $\alpha \in [0, 1]$.

   The rational behind this loss term is that the modes we wish to disentangle are the eigenvectors of $\mathcal{K}$. We want to constrain some of the eigenvalues (subset $S$) to be close to 1 (these will relate to the static modes) and the



* Icons created by iconsmind.com, Oleksandr Panasovskyi, Lucas Rathgeb, and ahmadwil from Noun Project

Figure 1: An architecture diagram of SKD

rest of eigenvalues (subset $D$) to have real parts which do not exceed $\alpha$ (these will relate to the dynamic modes). Interested readers looking for a more comprehensive explanation should consult the SKD paper [4].

For the combined loss term, we assign weights to each of the aforementioned loss terms:

$$\mathcal{L} = w_{\text{pred}_Z}\mathcal{L}_{\text{pred}_Z} + w_{\text{pred}_X}\mathcal{L}_{\text{pred}_X} + w_{\text{rec}}\mathcal{L}_{\text{rec}} + w_{\text{eig}}\mathcal{L}_{\text{eig}} \tag{5}$$

In Figure 1 we show an architecture diagram of SKD using the notations above.

#### 2.2.2 Reimplementation

During our experiments on SKD we encountered a few issues (which, as of date, have not yet been fixed in the official repository) that prevented us from reproducing the results reported in the SKD paper. Hence, we propose our reimplementation of SKD which resolves these issues:

1. We introduce a fix to the dimension mismatches in the source code of the encoder and decoder. The issue is that, in the original implementation of SKD, the usage of hyperparameters $K$ and $\mathcal{H}$ (dimension of a hidden layer) is wrong, and not as described in Table 5 in the SKD paper. Thus, in a few lines we swap between hyperparameters $K$ and $\mathcal{H}$ in order to fit Table 5 in the SKD paper, which reports the correct architecture details.
2. In the original implementation of SKD, in different batches the subset $S$ may actually be of different sizes. This is since, as an intentional implementation detail, the algorithm that delimits $S$ (the method *get_unique_num()*) considers a pair of conjugate eigenvalues as a single entity. While this approach may be justified, we found that this inconsistency in the size of $S$, which is too batch-dependent, makes the convergence of the spectral loss term, as well as the reproduction, harder, since the edge eigenvalues may be considered as members of $S$ and $D$ interchangeably between batches, such that the static and dynamic terms that are added in the spectral loss term (Equation 4) compete over the edge eigenvalues in such a way that two subsequent epochs often yield checkpoints such that one has good disentanglement results while the other has poor disentanglement results. We solve this by removing *get_unique_num()* completely. Instead, we always consider the $s$ eigenvalues closest to $1$ as static-related, regardless of whether they are parts of conjugate pairs or not.

3. In the original implementation of SKD, depending on the choice of random seed and hyperparameters, the training often ($\geq 40\%$ of the runs in our experiments) fails due to NaN values in the gradients. This issue is not sustainable for reproducible research. To avoid this issue, we apply gradient clipping.
4. In contrast to the original implementation of SKD, we use float64 operations in the Koopman module and the spectral loss calculation, as it is the suggested practice for working with eigendecompositions, since it provides a better numerical stability.
5. In contrast to the original implementation of SKD which lacked a learning rate scheduler, to achieve better convergence we use a learning rate scheduler that decays the learning rate on plateau.
6. The value of $\mathcal{H}$ hyperparameter reported in Table 6 in the SKD paper for the Sprites dataset (will be presented later) seems to be incorrect. We were able to reproduce the results on this dataset using $\mathcal{H} = 80$. Similarly, the value of $s$ hyperparameter reported in Table 6 in the SKD paper for the Sprites dataset seems to be incorrect. We were able to reproduce the results on this dataset using $s = 7$, and after applying the change described in item 2 on this list using $s = 12$.

# 3 Related Work

## 3.1 Sequential Disentanglement

Three meta-approaches can be seen in the existing literature on sequential two-factor disentanglement learning: **static and dynamic** factorization [21], **global and local** factorization [45; 47], and **seasonal and trend** factorization [48].

A limited amount of works have so far been performed on sequential multifactor disentanglement learning:

1. Decompositional Disentangled Predictive Autoencoder (DDPAE) addresses the challenge of predicting future video frames by decomposing high-dimensional video data into components and disentangling them into low-dimensional temporal dynamics, achieving unsupervised recovery of underlying components in complex datasets [13].
2. Multi-Disentangled-Features Gaussian Processes Variational Autoencoder (MGP-VAE) uses Gaussian processes to model latent spaces for unsupervised learning of disentangled static and dynamic features in video sequences, incorporating a novel geodesic loss to capture data manifold curvature, improving video prediction performance [5].
3. Factorizing Variational Autoencoder (FAVAE) leverages the information bottleneck principle to disentangle dynamic factors with similar temporal dependencies in sequential data, without the need for prior modeling, proving effective across video, speech, and synthetic datasets [49].
4. SKD, a Koopman-based autoencoder network, introduces a spectral loss to enforce structured disentanglement in multifactor representations, enabling advanced manipulation of individual factors and showing significant improvements over existing methods on benchmark tasks [4].

In the domain of audio and music both non-sequential and sequential disentanglement learning models have been employed to tackle real-world tasks such as timbre transfer, style transfer, and voice conversion [15; 16; 24; 12; 26; 25; 42; 22; 27; 43; 28]. These works introduced disentanglement learning (e.g., timbre and pitch factorization) as an instrument to a broader area of research on these tasks, which have often been studied without enforcing learning specifically disentangled representations [14; 30; 6; 1; 10]. Evidently, incorporating disentanglement learning to tackle these tasks has led to the development of more interpretable and controllable models.

A main challenge in disentanglement learning is the **shortcut problem** [41], also referred to as the **disentanglement-reconstruction tradeoff** [20]. If the latent space has too many dimensions, the decoder tends to disregard the latent variables linked to the intended factors of variation, instead relying on the capacity of the nuisance variables to perform the reconstruction. Conversely, if the latent space has too few dimensions, the decoder is forced to rely on the specified variables but is restricted in the amount of information it can use, leading to a more distorted reconstruction compared to the autoencoder's input.

A recent work that follows the static and dynamic factorization approach to sequential disentanglement noted the lack of a formal definition of what constitutes a static factor and what constitutes a dynamic factor. They proposed such a formal definition, and by doing so they also unveiled that static factors may causally influence the dynamic ones, e.g., a single class of a dynamic factor may appear as two different classes of the dynamic factor given two different classes of a static factor [39].

## 3.2 Koopman Theory and DMD

### 3.2.1 Core Research

The Koopman theory and DMD techniques offer a framework for decomposing complex dynamics into a set of modes with specific oscillation frequencies and growth/decay rates, in a similar fashion to the way in which the Fourier transform decomposes a signal into its constituent frequencies.

Koopman's work laid the foundation by demonstrating that nonlinear dynamics could be represented using linear transformations in Hilbert space, extending classical mechanics to this operator-theoretic perspective [19]. In recent years, Koopman spectral theory has gained prominence by representing nonlinear dynamics as infinite-dimensional linear operators, enabling predictions and control of nonlinear systems through linear methods [7].

DMD, introduced within the fluid mechanics community, complements Koopman theory by providing numerical techniques to extract dominant dynamic modes from experimental or numerical data without requiring a model equation. Early DMD work demonstrated its ability to identify coherent structures in fluid flows, enabling the reduction of large-scale systems into lower-dimensional dynamical representations [36; 35].

Recent advancements in DMD theory have expanded its applicability, offering new sampling strategies, noise mitigation techniques, and strengthening its connections with Koopman theory and other methods, such as the eigensystem realization algorithm (ERA) and linear inverse modeling (LIM) [46].

Together, these techniques provide powerful tools for data-driven analysis of nonlinear dynamical systems across a variety of fields.

### 3.2.2 Software Tools

Several software tools that leverage Koopman theory and DMD for modeling and predicting complex system dynamics have recently been developed:

1. DLKoopman is a deep learning-based package that encodes nonlinear dynamical systems into linear spaces while simultaneously learning the linear dynamics, offering a generalized approach to predicting both individual states and trajectories [11].
2. PyKoopman provides a framework for approximating the Koopman operator, enabling system identification for both unforced and actuated systems, and builds upon equation-free DMD techniques for predicting nonlinear dynamics using linear systems theory [32].
3. PyDMD implements DMD and its variants, expanding the method's ability to handle noisy, multiscale, high-dimensional, and nonlinear dynamics [17].

These offer comprehensive documentation and practical coding examples, and serve as powerful tools for researchers analyzing dynamical systems across various fields.

### 3.2.3 Application

Recent works have advanced the application of Koopman theory to time series modeling, addressing challenges posed by nonlinear dynamics, distribution shifts, and long-term predictions:

1. Consistent Koopman Autoencoder introduces a novel framework that leverages both forward and backward dynamics to provide robust and accurate long-term predictions, especially for high-dimensional systems [2].

2. Koopman Neural Forecaster (KNF) addresses distributional shifts in time series by using global and local Koopman operators to model changing dynamics and improve resilience against non-stationarity [47].
3. Koopman Invertible Autoencoder (KIA) further enhances long-term prediction accuracy, achieving significant improvements in noisy environments such as pendulum and climate datasets [44].
4. To tackle the complexities of non-stationary time series, Koopa employs Fourier Filter to disentangle time-invariant and time-varying components, and uses Koopman Predictors to extend forecasting horizons efficiently, while also optimizing for training time and memory usage [23].
5. Koopman Variational Autoencoder (KoVAE) introduces a novel generative framework for time series data, combining the stability of variational autoencoders [18] with the spectral tools of Koopman theory to generate realistic sequences, outperforming existing generative models across various benchmarks [31].

These methods demonstrate Koopman theory's growing importance in modeling and forecasting nonlinear dynamic time series data.

#### 3.2.4 Current State of Research

Koopman methods are increasingly favored for disentangled representation learning due to their ability to linearize nonlinear dynamics and provide structured, interpretable latent spaces. These methods exploit desired properties such as decomposition to eigenfunctions representing different dynamic modes, separation of time-invariant and transient dynamics, and spectral interpretability, enabling precise decomposition of complex systems into independent components, and allowing robust explainability and control. Unlike traditional black-box approaches, Koopman-based methods offer theoretical foundations, enhanced generalization, reduced redundancy, and the ability to reconstruct and predict system behavior efficiently. Their growing importance is driven by their utility in extracting actionable insights from high-dimensional, time-series data in fields like robotics, control, and physics-informed machine learning.

Koopman theory has been applied in representation learning to extract disentangled latent variables by leveraging its ability to linearize nonlinear dynamics. Applications often utilize Koopman eigenfunctions to separate state dynamics into invariant subspaces, enabling structured representations in robotics, control, and physics-informed machine learning. While progress has been made with methods such as DMD and Koopman autoencoders, challenges remain in generalizing these techniques to high-dimensional, noisy, or coupled systems. Theoretical gaps persist in achieving complete disentanglement of latent representations, particularly in identifying and leveraging time-invariant modes.

Reducing all time-invariant dynamic modes to a single mode extends current knowledge by offering a more compact representation of invariants while retaining critical system properties. By assigning one eigenvector to a single time-invariant mode, the method ensures inherent orthogonality - a desirable property for disentangled representations. This reduction simplifies latent space dynamics, enhances interpretability, allows introduction of novel constraints on representation of invariants, and improves computational efficiency by focusing on transient dynamics. Such an approach addresses redundancy in existing methods and provides a unified invariant descriptor, paving the way for more efficient applications in dynamical systems analysis and control.

# 4 Method

## 4.1 Single Static Mode Structured Koopman Disentanglement (SSM-SKD)

### 4.1.1 Motivation

Since according to the design of SKD we constrain all static modes to have eigenvalues which are close to $1$, we may also consider having a single static mode with an eigenvalue which is close to $1$. This may possibly aid in introducing further constraints on the static modes, which are now reduced to a single static mode. Therefore, we call this model **SSM-SKD**.

### 4.1.2 Architecture in Relation to SKD

We start by defining $s = 1$, and try to reproduce the results of SKD using this setting. We then encounter the shortcut problem: if $K$ is small ($K \leq 8$) we are unable to achieve SKD's reconstruction; otherwise we are unable achieve SKD's static-dynamic disentanglement, since it makes it easier for the model to use the higher capacity of all other modes to encode static information.

We solve this by changing the approximation of Koopman operator from being performed per-batch to being performed per-instance. In order to do this, we compute a solution to the least squares problem for the linear system $Z_{i,0:T-2}\mathcal{K}_i = Z_{i,1:T-1}$ for each $0 \leq i \leq N-1$. Hence, the $N$ resulting matrices $\mathcal{K}_i$ are the approximated Koopman operators of instances of the batch. We redefine $\mathcal{K}$ for SSM-SKD to be the tensor concatenating all $\mathcal{K}_i$ matrices along a new dimension at dimension index $0$.

### 4.1.3 Latent Space Extraction in Relation to SKD

SKD extracts a latent space per batch. Consistent with DMD, SKD considers the result of multiplying a latent matrix $Z_i$, where $i$ is the index of instance in batch, by a submatrix of the eigenvector matrix of $\mathcal{K}$ which consists of the desired modes as the Koopman-latent representation of the instance along these modes. Thus, in order to perform attribute swapping between two instances, we separately multiply their latent matrices by the eigenvector matrix, and then we perform a swap on the resulting product matrices along the desired modes by which we want to do so. Then, following a multiplication of the swapped matrices by the inverse of eigenvector matrix we get the new latent matrices, which we now can pass through the decoder.

Since in SSM-SKD we approximate the Koopman operator per-instance, we cannot use SKD's latent space extraction method. Instead, we multiply each latent matrix of the instances by the submatrix of corresponding eigenvector matrix which consists of the static mode, then by the corresponding submatrix of inverse of corresponding eigenvector matrix, to get the static latent representations of instances. Similarly, we multiply each latent matrix of the instances by the submatrix of corresponding eigenvector matrix which consists of the dynamic modes, then by the corresponding submatrix of inverse of corresponding eigenvector matrix, to get the dynamic latent representations of instances. We then treat the $K$ feature coordinates of static latent representation and $K$ feature coordinates of dynamic latent representation of an instance as its static and dynamic channels, respectively. In order to perform attribute swapping we can now simply swap the contents at the desired channels between two instances, then for each instance pass the sum of its new static and dynamic latent representations through the decoder.

We visualize SKD's and SSM-SKD's latent space extraction and swap methods in Figure 2. Note that, different from SKD's method, SSM-SKD's alternative method is currently not based on any theoretical guarantee or justification.

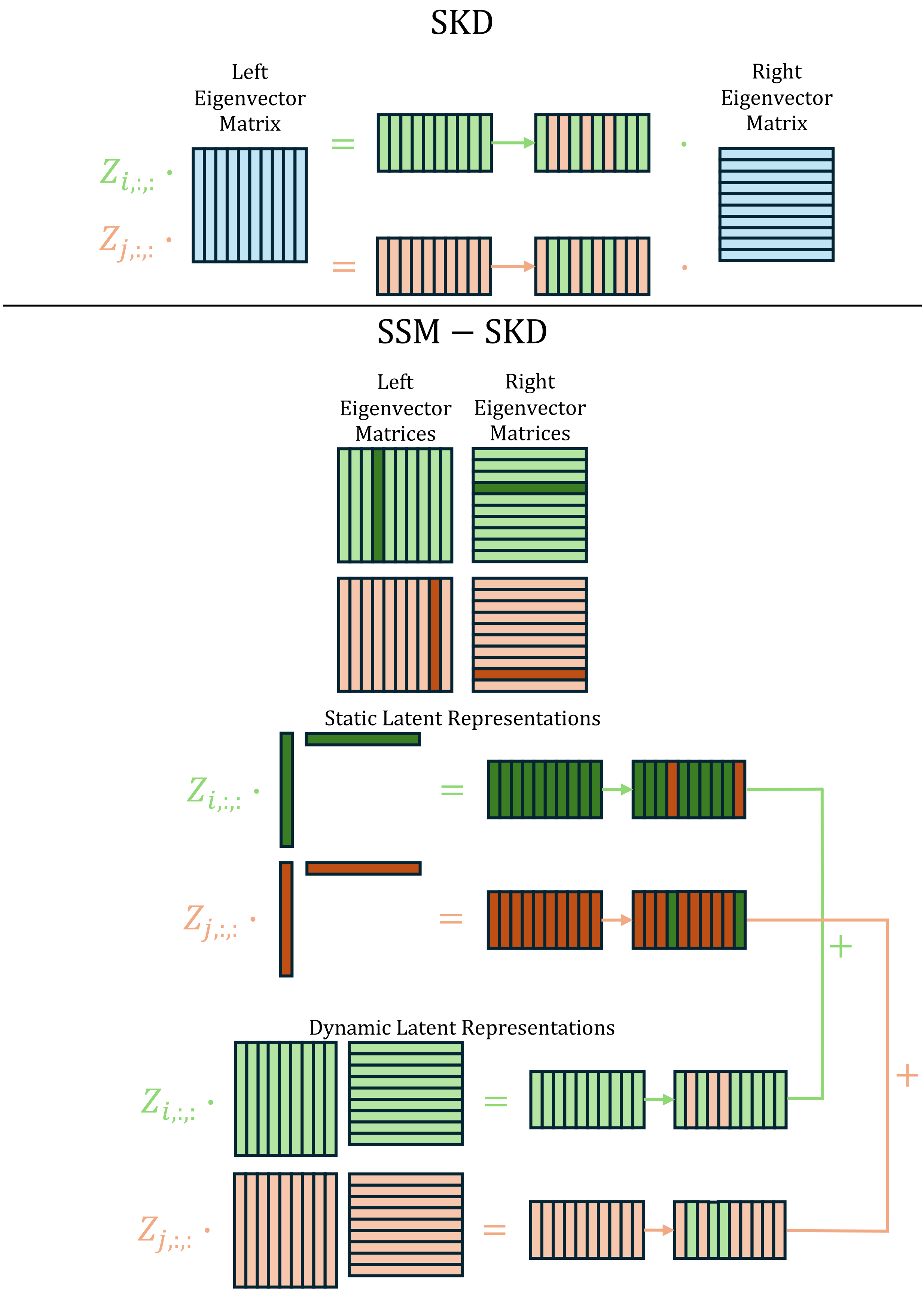


Figure 2: Latent space extraction and swap in SKD and in SSM-SKD

#### 4.1.4 Latent Space Exploration in Relation to SKD

In order to perform latent space exploration we train a classifier for static factors of the desired dataset.

In order to identify which static modes relate to each factor, SKD employs a brute-force search over the power set of static modes.

We propose a greedy latent space exploration algorithm as follows:

1. For each static factor, initialize an empty list.
2. For each test instance, swap all of its static channels with those of a random test instance from the same batch.
3. Calculate the accuracy of each static factor against the original label. We call this the reference accuracy of the factor.
4. For each static channel $i$:
    (a) For each test instance, swap all of its static channels except $i$ with those of a random test instance from the same batch.
    (b) Calculate the accuracy of each static factor against the original label.
    (c) If the maximum of differences between these accuracies and the reference accuracies among the static factors is greater than $0$, insert $i$ to the list related to the static factor for which the difference is maximal. Otherwise, disregard it.

Note that, for statistical robustness, we consider $25$ repetitions of the test split as the actual test split for this algorithm, and we use a batch size of $256$.

Since each static channel is a coordinate rather than a vector, all static channels are readily orthogonal to each other. Therefore, this partition of the channels is optimal in regards to the maximality of sum of accuracies of static factors calculated against the original labels after swapping all static channels except those in the corresponding subsets.

We prove this as follows. Let $P$ be the partition we got from the algorithm and $O$ be the optimum partition. If $P$ is different from $O$, then there exists at least one static channel that is assigned to different subsets in $P$ and $O$. If this static channel is assigned to the disregarded subset in partition $P$, then we can further optimize $O$ by assigning it to the disregarded subset, since it improves no accuracy, but $O$ is already optimized. Otherwise, it follows that this static channel is assigned in $O$ to the subset that corresponds to a static factor for which the difference described above is lower than or equal to the maximum difference. If it is lower, then we can optimize $O$ by assigning this static channel to the subset that corresponds to the static factor for which the difference is maximal, but $O$ is already optimized. Otherwise, repeat for each of the rest of static channels which are assigned to different subsets in $P$ and $O$. Then the sum of accuracies according to $O$ is equal to the sum of accuracies according to $P$, hence we find that $P$ is indeed optimal.

Note that, while this algorithm yields an optimal solution in regards to the sum of accuracies, it does not minimize leakage between factors.

## 4.2 Datasets

### 4.2.1 Sprites

The **Sprites** dataset [21] consists of video frame sequences of length $8$ and frame size $64 \times 64$. The factors in this dataset are:

| **Name** | **Type** | **Number of Classes** |
|---|---|---|
| Skin | static | 6 |
| Pants | static | 6 |
| Top | static | 6 |
| Hairstyle | static | 6 |
| Action | dynamic | 9 |

Table 1: Factors in the Sprites dataset

There are overall $11664$ instances. Train/test split is $77\%/23\%$.

### 4.2.2 dSprites (Our Variant)

Our first variant of the dSprites dataset [29], which we refer to as **dSprites** in this work, consists of video frame sequences of length $8$ and frame size $64 \times 64$. The factors in this dataset are:

| **Name** | **Type** | **Number of Classes** |
|---|---|---|
| Color | static | 6 |
| Shape | static | 3 |
| Position X | static | 8 |
| Position Y | static | 8 |
| Scale | dynamic | 10 |

Table 2: Factors in the dSprites dataset

There are overall $11520$ instances. Train/test split is $80\%/20\%$.

### 4.2.3 Moving dSprites (Our Variant)

Our second variant of the dSprites dataset, which we refer to as **Moving dSprites** in this work, consists of video frame sequences of length $8$ and frame size $64 \times 64$. The factors in this dataset are:

| **Name** | **Type** | **Number of Classes** |
|---|---|---|
| Color | static | 6 |
| Shape | static | 3 |
| Scale | dynamic | 10 |
| Position X | dynamic | 8 |
| Position Y | dynamic | 8 |

Table 3: Factors in the Moving dSprites dataset

There are overall $11520$ instances. Train/test split is $80\%/20\%$.

#### 4.2.4 3D Shapes (Our Variant)

Our variant of the 3D Shapes dataset [8], which we refer to as **3D Shapes** in this work, consists of video frame sequences of length $10$ and frame size $64 \times 64$. The factors in this dataset are:

| **Name** | **Type** | **Number of Classes** |
|---|---|---|
| Floor Hue | static | 10 |
| Wall Hue | static | 10 |
| Object Hue | static | 10 |
| Shape | static | 4 |
| Initial Scale | augmentation | 6 |
| Scale | dynamic | 2 |
| Orientation | dynamic | 3 |

Table 4: Factors in the 3D Shapes dataset

There are overall $144000$ instances. Train/test split is $80\%/20\%$.

### 4.3 Metrics

#### 4.3.1 Static-Dynamic Disentanglement

Given that $F_{\text{s}}$ is the set of static factors, $F_{\text{d}}$ is the set of dynamic factors, $\text{Acc}(a, b)$ is the mean element-wise accuracy between two vectors $a$ and $b$, $\mathbb{C}_f(X)$ is the vector of classifications given by the classifier of factor $f$ to the instances in $X$, StaticSampleSwap$(X)$ and DynamicSampleSwap$(X)$ are the results of swap of static and dynamic factors of $X$ with those of a randomly weighted convex hull interpolation between $2$ permutations (by the batch dimension) of $X$, respectively, and $\mathcal{C}_f$ is the set of possible classes of factor $f$, we conceive our **static-dynamic disentanglement metric**:

$$
\begin{aligned}
\mathcal{D}_{\text{sd}}(X) = 1 - \frac{1}{2(|F_{\text{s}}| + |F_{\text{d}}|)} \Bigg( &\sum_{f \in F_{\text{s}}} \left| \text{Acc}(\mathbb{C}_f(\text{StaticSampleSwap}(X)), \mathbb{C}_f(X)) - \frac{1}{|\mathcal{C}_f|} \right| + \\
&\sum_{f \in F_{\text{d}}} \left| \text{Acc}(\mathbb{C}_f(\text{StaticSampleSwap}(X)), \mathbb{C}_f(X)) - 1 \right| + \\
&\sum_{f \in F_{\text{s}}} \left| \text{Acc}(\mathbb{C}_f(\text{DynamicSampleSwap}(X)), \mathbb{C}_f(X)) - 1 \right| + \\
&\sum_{f \in F_{\text{d}}} \left| \text{Acc}(\mathbb{C}_f(\text{DynamicSampleSwap}(X)), \mathbb{C}_f(X)) - \frac{1}{|\mathcal{C}_f|} \right| \Bigg)
\end{aligned}
\tag{6}
$$

The intuition behind this metric is that in each swap the swapped factors should yield accuracies that correspond to uniform distributions (since our datasets are uniformly distributed across all factors), while the fixed factors should yield accuracies of $1$.

We calculate this metric on the test split using a batch size of $256$ over $25$ epochs of evaluation.

#### 4.3.2 Multifactor Disentanglement

The scope of multifactor disentanglement in this work is disentanglement of static factors. By using the same intuition described above, given that $\text{FactorialSampleSwap}_f(X)$ is the result of swap of all factors of $X$ except $f$ with those of a randomly weighted convex hull interpolation between 2 permutations (by the batch dimension) of $X$, we conceive our **multifactor disentanglement metric**:

$$\mathcal{D}_{\text{mf}}(X) = 1 - \frac{1}{|F_{\text{s}}|(|F_{\text{s}}| + |F_{\text{d}}|)} \left( \sum_{f \in F_{\text{s}}} \sum_{g \in F_{\text{s}} \cup F_{\text{d}}} \begin{cases} \left|\text{Acc}(\mathbb{C}_g(\text{FactorialSampleSwap}_f(X)), \mathbb{C}_g(X)) - 1\right|, & \text{if } g = f \\ \left|\text{Acc}(\mathbb{C}_g(\text{FactorialSampleSwap}_f(X)), \mathbb{C}_g(X)) - \frac{1}{|\mathcal{C}_g|}\right|, & \text{otherwise} \end{cases} \right) \tag{7}$$

We calculate this metric on the test split using a batch size of $256$ over $25$ epochs of evaluation as well.

### 4.4 Hyperparameter Tuning

For all datasets we globally fix the following hyperparameters:

| **Hyperparameter** | **Value** |
|---|---|
| Random seed | $7253$ |
| Learning rate | $0.001$ |
| $N$ | $32$ |
| Gradient clipping maximum norm | $5$ |
| $w_{\text{rec}}$ | $16$ |

Table 5: Values of global hyperparameters

For each dataset we perform a grid search of $200$ random combinations of the following hyperparameters from the given ranges:

| **Hyperparameter** | **Range** |
|---|---|
| $K$ | $[8, 40]$ |
| $\mathcal{H}$ | $[80, 200]$ |
| $w_{\text{pred}_Z} = w_{\text{pred}_X}$ | $[0.25, 4]$ |
| $w_{\text{eig}}$ | $[0.25, 4]$ |
| $\alpha$ | $[0.125, 0.875]$ |

Table 6: Ranges of random hyperparameters

The following table lists all dataset-specific hyperparameters of best models:

| **Hyperparameter** | **Sprites** | **dSprites** | **Moving dSprites** | **3D Shapes** |
|---|---|---|---|---|
| Number of epochs | $250$ | $250$ | $250$ | $75$ |
| $K$ | $15$ | $24$ | $28$ | $28$ |
| $\mathcal{H}$ | $170$ | $200$ | $160$ | $200$ |
| $w_{\text{pred}_Z} = w_{\text{pred}_X}$ | $1$ | $4$ | $4$ | $4$ |
| $w_{\text{eig}}$ | $1$ | $4$ | $4$ | $4$ |
| $\alpha$ | $0.75$ | $0.125$ | $0.875$ | $0.875$ |

Table 7: Values of dataset-specific hyperparameters of best models

# 5 Results

## 5.1 Presentation

Results are presented first through tables that show individual accuracies for each factor after performing static/dynamic sample swap (in static-dynamic disentanglement) or factorial sample swap (in multifactor disentanglement). The order of columns from left to right is swap type, static factors, and dynamic factors. In bold - accuracies that are desired to be 1, while the others are desired to reflect uniform distribution with respect to the number of possible classes of each factor. Then, the corresponding results of the metrics defined above are presented in the last table of each subsection.

## 5.2 Static-Dynamic Disentanglement

| **Swap** | **Skin** | **Pants** | **Top** | **Hairstyle** | **Action** |
|---|---|---|---|---|---|
| Static | 0.1861 | 0.1649 | 0.1732 | 0.1766 | **0.9946** |
| Dynamic | **0.9607** | **0.9878** | **0.9942** | **1** | 0.1388 |

Table 8: Accuracies of factors after static and dynamic sample swaps on Sprites

| **Swap** | **Color** | **Shape** | **Position X** | **Position Y** | **Scale** |
|---|---|---|---|---|---|
| Static | 0.1859 | 0.364 | 0.1443 | 0.1411 | **0.712** |
| Dynamic | **0.9919** | **0.9256** | **0.9996** | **0.9993** | 0.1518 |

Table 9: Accuracies of factors after static and dynamic sample swaps on dSprites

| **Swap** | **Color** | **Shape** | **Scale** | **Position X** | **Position Y** |
|---|---|---|---|---|---|
| Static | 0.1902 | 0.8488 | **0.8857** | **0.9827** | **0.9939** |
| Dynamic | **0.9976** | **0.4907** | 0.1301 | 0.1721 | 0.1605 |

Table 10: Accuracies of factors after static and dynamic sample swaps on Moving dSprites

| **Swap** | **Floor Hue** | **Wall Hue** | **Object Hue** | **Shape** | **Scale** | **Orientation** |
|---|---|---|---|---|---|---|
| Static | 0.102 | 0.1176 | 0.1386 | 0.3787 | **0.9249** | **0.9975** |
| Dynamic | **0.968** | **0.9995** | **0.9521** | **0.8836** | 0.6122 | 0.3693 |

Table 11: Accuracies of factors after static and dynamic sample swaps on 3D Shapes

| **Sprites** | **dSprites** | **Moving dSprites** | **3D Shapes** |
|---|---|---|---|
| 0.9872 | 0.9491 | 0.8699 | 0.9492 |

Table 12: Static-dynamic disentanglement metric results of SSM-SKD

## 5.3 Multifactor Disentanglement

| **Retain** | **Skin** | **Pants** | **Top** | **Hairstyle** | **Action** |
|---|---|---|---|---|---|
| Skin | **0.8425** | 0.1769 | 0.1828 | 0.1735 | 0.1293 |
| Pants | 0.189 | **0.9853** | 0.1747 | 0.1658 | 0.1164 |
| Top | 0.1755 | 0.1689 | **0.9924** | 0.1659 | 0.115 |
| Hairstyle | 0.1862 | 0.1667 | 0.1714 | **0.9845** | 0.116 |

Table 13: Accuracies of factors after factorial sample swaps on Sprites

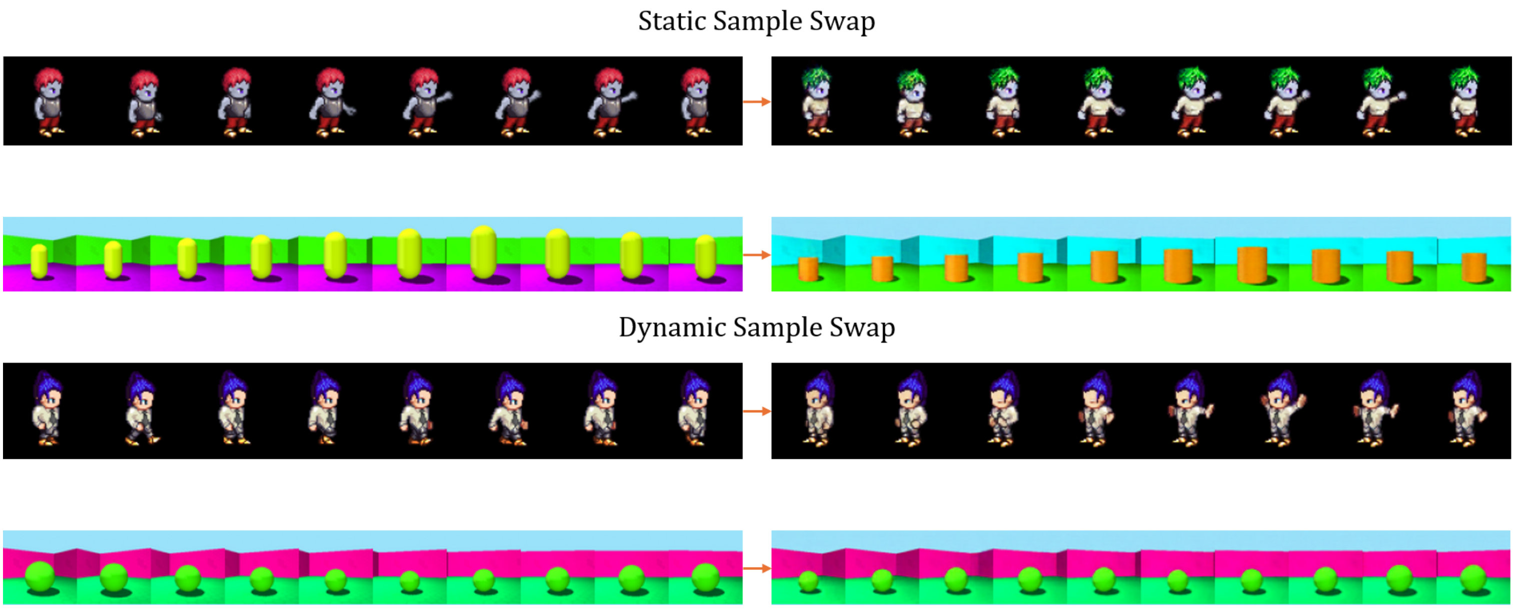


Figure 3: Static and dynamic sample swap results of SSM-SKD on Sprites and 3D Shapes

| Retain | Color | Shape | Position X | Position Y | Scale |
|---|---|---|---|---|---|
| Color | **0.9846** | 0.3438 | 0.1366 | 0.1378 | 0.1022 |
| Shape | 0.1736 | **0.589** | 0.1428 | 0.1624 | 0.1375 |
| Position X | 0.172 | 0.3469 | **0.6748** | 0.2695 | 0.1041 |
| Position Y | 0.17 | 0.4039 | 0.1753 | **0.4654** | 0.1009 |

Table 14: Accuracies of factors after factorial sample swaps on dSprites

| Retain | Color | Shape | Scale | Position X | Position Y |
|---|---|---|---|---|---|
| Color | **0.9845** | 0.3431 | 0.1043 | 0.1301 | 0.1315 |
| Shape | 0.18 | **0.4738** | 0.1268 | 0.1485 | 0.1465 |

Table 15: Accuracies of factors after factorial sample swaps on Moving dSprites

| Retain | Floor Hue | Wall Hue | Object Hue | Shape | Scale | Orientation |
|---|---|---|---|---|---|---|
| Floor Hue | **0.9762** | 0.1056 | 0.1051 | 0.2594 | 0.5032 | 0.3378 |
| Wall Hue | 0.1079 | **0.9867** | 0.1038 | 0.2617 | 0.5018 | 0.3461 |
| Object Hue | 0.1026 | 0.1047 | **0.8094** | 0.3017 | 0.5 | 0.3353 |
| Shape | 0.122 | 0.1155 | 0.1462 | **0.7882** | 0.5254 | 0.353 |

Table 16: Accuracies of factors after factorial sample swaps on 3D Shapes

| Sprites | dSprites | Moving dSprites | 3D Shapes |
|---|---|---|---|
| 0.9836 | 0.9142 | 0.9348 | 0.971 |

Table 17: Multifactor disentanglement metric results of SSM-SKD

# 6 Discussion

## 6.1 Comparison with SKD on Sprites

### 6.1.1 Metrics

As we were able to reproduce, the static-dynamic disentanglement metric result of SKD on Sprites is $0.9981$, so SSM-SKD's corresponding result is $0.0109$ lower. However, using the partial information in Table 1 in the SKD paper, we get that the multifactor disentanglement metric result of SKD on Sprites is $0.9276$, so SSM-SKD's corresponding result is $0.056$ higher, which is a much more significant difference.

### 6.1.2 Model Size

Both SKD and SSM-SKD use about $2$ million parameters when trained on Sprites.

### 6.1.3 Latent Space Size

Whereas in SKD $K = 40$ when trained on Sprites, in SSM-SKD $K = 15$ when trained on Sprites, so the latent space is $2.667$x more compressed in SSM-SKD.

### 6.1.4 Spectrum of the Koopman Matrices

Since in SSM-SKD, unlike in SKD, we are dealing with underdetermined systems ($K$ is larger than the number of prediction frames, which is $T-1$), the solutions to the least squares problems are, as an implementation detail, minimum norm solutions. We observe that, indeed, in SSM-SKD $K - T + 1$ eigenvalues of the Koopman matrices are always $0$. Similarly, the latent prediction loss term is always $0$.

It is interesting that on Sprites, for example, $K = 7$ is not enough to achieve the results that we get when $K = 15$, since $8$ of the dynamic modes are not in use. This might be explained by the fact that the modes that are in use are more expressive (the eigenvectors are longer) as $K$ is larger.

## 6.2 Defining Static and Dynamic Factors

### 6.2.1 Our Definition

Even though physical dynamics (displacement, velocity, acceleration, rotation, etc.) can be considered as dynamic factors in sequences, it is possible to think of ways to consider them as static factors. For example, we can consider the velocity component of a constant velocity motion sequence as a static factor, since the velocity does not change throughout the sequence. This shows that there is a need in a universal definition of static and dynamic factors.

In addition to the formal definition proposed in [39], we want to give a more intuitive definition to static and dynamic factors.

A **static factor** is a factor that, when given by one sequence element, can be inferred the same on all rest of sequence elements. Thus, a **dynamic factor** is a factor that, when given by one sequence element, does not appear the same on all rest of sequence elements.

In relation to the example given above, this definition prevents considering constant velocity as a static factor, since we cannot infer velocity from one sequence element alone.

### 6.2.2 Orientation in Sprites

For an unknown reason, previous works considered the orientation of the characters in Sprites as part of the action factor (a dynamic factor), so we did the same, but, as pointed out in [39], since the orientation does not change throughout the sequences, it is actually a static factor.

### 6.2.3 Initial Scale in 3D Shapes

The two possible scale dynamics in 3D Shapes are grow first and shrink first. The initial scale factor is an augmentation. Since initial scale only relates to the first sequence element, it can neither be considered static nor dynamic. What can be done in order to include this factor in the metrics is to define the scale factor in such a way that assigns different classes to all different combinations of initial scale and scale dynamics.

## 6.3 Towards Standardizing Reproducible Research in Machine Learning via Comprehensive Reporting of Environment Details

While conducting this research we encountered a number of issues related to reproducibility. We found that, even though reproducible research in machine learning is of growing interest [40; 37; 38], the existing literature does not give enough emphasis to reporting of environment details. Since these may affect reproductions, and guaranteeing reproducibility across different hardware is a work in progress in many software platforms, we deem it necessary to report them. For this reason we propose a standard for environment details that should be reported in every empirical machine learning research paper. Interested readers should refer to Appendix A for our environment details report, which also serves as an example for others to follow. Note that each such report should include the equivalent details when using other sets of tools (Windows instead of Linux, Java instead of Python, conda instead of pip, TensorFlow instead of PyTorch, etc.).

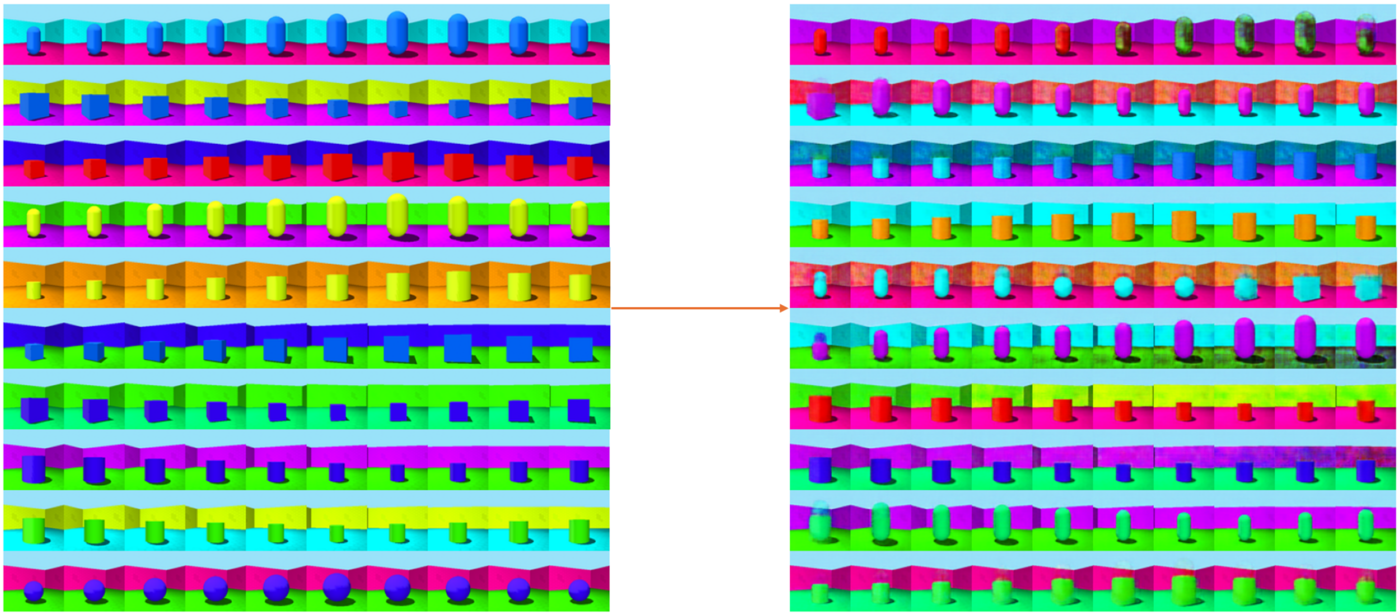


Figure 4: Inconsistency between frames after static sample swap

# 7 Limitations

## 7.1 Theoretical Gaps

As we pointed out earlier, our latent space extraction method lacks a theoretical justification. Contrary to SKD's extraction method, it is unclear why our coordinate-based extraction works.

## 7.2 Dataset Splits

Our evaluation in this work is based on an existing foul methodology which uses train and test splits with no validation splits. For this reason, we used the test splits as validation splits as well for model selection, which means that our results may suffer from overfitting to the test splits.

## 7.3 Inconsistency Between Frames

As can be seen in Figure 4, after performing static sample swap (though this artifact is not limited to static sample swap), static factors may be inconsistent between frames. Since our classifier infers the classes of the factors from the entire sequences, our metric results do not necessarily reflect this issue. Thus, in order to evaluate the consistency between frames, a per-frame classifier is needed, as well as a different metric for consistency of the method.

Regarding this issue, we found that disabling the LSTM layers of the model results in even worse inconsistency between frames. This finding corroborates the assumption that the LSTM layers are playing an integral part in both SKD and SSM-SKD.

## 7.4 Poor Performance on dSprites Variants

As can be seen in Table 14 and Table 15, our model does not perform well on both dSprites and Moving dSprites. The case is worse on Moving dSprites, where our model fails to disentangle the shape from the dynamics. We also experimented on a variant of dSprites which includes an orientation dynamic factor, and our model did not perform well on it as well. A possible explanation for the poor performance on this variant was given in [39]. In essence, the different symmetries of the shapes render the orientation dynamics dependent on the shapes.

# 8 Future Work

## 8.1 Disentanglement of Dynamic Factors

Disentanglement of dynamic factors is not in the scope of this work. However, a recent preliminary experimentation we performed on 3D Shapes revealed that our method may perform it well.

## 8.2 Ablation Study on Our Proposed Latent Space Extraction Method

A recent preliminary experimentation we performed showed that it is likely that the improvement in multifactor disentanglement in our method in relation to SKD is mainly a result of our latent space extraction method rather than the architectural change. Thus, it is desirable to perform an ablation study that will compare SSM-SKD to SKD with our latent space extraction method, and possibly with our latent space exploration method as well, across all four datasets.

## 8.3 Introducing Constraints on the Static Latent Representation

Current deep learning software platforms do not support backpropagation through eigenvectors due to numerical difficulties. It is, however, possible to approximate the corresponding eigenvector of a real eigenvalue in a backpropagation-friendly manner by employing the shifted inverse power method (SIPM). Since our static latent representation relies solely on a real eigenvalue which is close to $1$, we can use this to introduce new constraints (e.g., contrastive, mutual information-based) on our static latent representation. Our preliminary experimentation shows that this direction of research is indeed possible, though it is computationally inefficient, and there is also a need to perform a theoretical study on the validity of usage of the normalized approximation of corresponding right eigenvector given by applying SIPM on the transpose of Koopman matrix.

## 8.4 Sequential Multifactor Disentanglement Benchmark

Due to the unaligned existing literature on sequential multifactor disentanglement, we deem it necessary to establish a sequential multifactor disentanglement benchmark which is composed of controllable datasets with train/validation/test splits and robust metrics.

## 8.5 Extended Sprites Dataset

Since results on the current Sprites dataset are nearly saturated, and more similar data is available through the Liberated Pixel Cup Spritesheet Character Generator (available at https://github.com/sanderfrenken/Universal-LPC-Spritesheet-Character-Generator), we propose to create an extended Sprites dataset that consists of more static and dynamic factors and classes and correctly defines orientation as a static factor.

## 8.6 Failure Case Study on dSprites Variants

We propose to study the failure cases of our model on dSprites variants. Particularly, a careful incremental regeneration of Moving dSprites should assist in identifying the breaking point of the model.

## 8.7 Beyond Visual Sequential Multifactor Disentanglement

It is desirable to expand sequential multifactor disentanglement research beyond the visual modality. An interesting possibility is to test our model against dMelodies, a symbolic music dataset for sequential multifactor disentanglement learning [33]. Additionally, it is possible to create an equivalent real-world raw waveform music dataset using static and dynamic audio descriptor extraction tools, such as the Timbre Toolbox [34].

## A Environment Details

| **Detail** | **Sprites / dSprites** | **Moving dSprites** | **3D Shapes** |
|---|---|---|---|
| CPU | Intel(R) Xeon(R) CPU E5-1650 v4 @ 3.60GHz | AMD EPYC 7343 16-Core Processor | Intel(R) Xeon(R) CPU E5-1650 v4 @ 3.60GHz |
| CPU architecture | x86_64 | x86_64 | x86_64 |
| Linux kernel version | 5.14.0-427.18.1 .el9_4.x86_64 | 5.14.0-427.20.1 .el9_4.x86_64 | 5.14.0-427.18.1 .el9_4.x86_64 |
| glibc version | 2.34 | 2.34 | 2.34 |
| GPU | NVIDIA GeForce RTX 3090 (Gigabyte Turbo) | NVIDIA GeForce RTX 3090 (Gigabyte Turbo) | NVIDIA GeForce RTX 3090 (Innovision INNO3D Blower) |
| NVIDIA driver version | 555.42.02 | 555.42.02 | 555.42.02 |
| NVIDIA VBIOS version | 94.02.26.08.1C | 94.02.42.40.34 | 94.02.42.00.02 |
| CUDA version | 12.1 | 12.1 | 12.1 |
| cuDNN version | 8.9.2 | 8.9.2 | 8.9.2 |
| Python version | 3.11.9 | 3.11.9 | 3.11.9 |
| pip version | 24.0 | 24.0 | 24.0 |
| NumPy version | 2.0.0 | 2.0.0 | 2.0.0 |
| PyTorch version | 2.3.1 | 2.3.1 | 2.3.1 |
| PyTorch Lightning version | 2.3.0 | 2.3.0 | 2.3.0 |
| Python package versions | requirements-exact.txt | requirements-exact.txt | requirements-exact.txt |

Table 18: Training environment details of our best models on each dataset

Listing 1: requirements-exact.txt

```
aiofiles==23.2.1
aiohttp==3.9.5
aiosignal==1.3.1
altair==5.3.0
annotated-types==0.7.0
anyio==4.4.0
arrow==1.3.0
attrs==23.2.0
boto3==1.34.128
botocore==1.34.128
bravado==11.0.3
bravado-core==6.1.1
certifi==2024.6.2
charset-normalizer==3.3.2
click==8.1.7
configparser==7.0.0
contourpy==1.2.1
```

```
cycler==0.12.1
dnspython==2.6.1
docker-pycreds==0.4.0
email_validator==2.1.2
fastapi==0.111.0
fastapi-cli==0.0.4
ffmpy==0.3.2
filelock==3.15.1
flake8==7.1.0
fonttools==4.53.0
fqdn==1.5.1
frozenlist==1.4.1
fsspec==2024.6.0
future==1.0.0
gin-config==0.5.0
gitdb==4.0.11
GitPython==3.1.43
gradio==4.36.1
gradio_client==1.0.1
h11==0.14.0
h5py==3.11.0
httpcore==1.0.5
httptools==0.6.1
httpx==0.27.0
huggingface-hub==0.23.4
idna==3.7
importlib_resources==6.4.0
isoduration==20.11.0
Jinja2==3.1.4
jmespath==1.0.1
jsonpointer==3.0.0
jsonref==1.1.0
jsonschema==4.22.0
jsonschema-specifications==2023.12.1
kiwisolver==1.4.5
lightning-utilities==0.11.2
markdown-it-py==3.0.0
MarkupSafe==2.1.5
matplotlib==3.9.0
mccabe==0.7.0
mdurl==0.1.2
```

```
monotonic==1.6
mpmath==1.3.0
msgpack==1.0.8
multidict==6.0.5
neptune==1.10.4
networkx==3.3
numpy==2.0.0
nvidia-cublas-cu12==12.1.3.1
nvidia-cuda-cupti-cu12==12.1.105
nvidia-cuda-nvrtc-cu12==12.1.105
nvidia-cuda-runtime-cu12==12.1.105
nvidia-cudnn-cu12==8.9.2.26
nvidia-cufft-cu12==11.0.2.54
nvidia-curand-cu12==10.3.2.106
nvidia-cusolver-cu12==11.4.5.107
nvidia-cusparse-cu12==12.1.0.106
nvidia-nccl-cu12==2.20.5
nvidia-nvjitlink-cu12==12.5.40
nvidia-nvtx-cu12==12.1.105
oauthlib==3.2.2
orjson==3.10.5
packaging==24.1
pandas==2.2.2
pillow==10.3.0
pip==24.0
platformdirs==4.2.2
prefigure==0.0.10
protobuf==5.27.1
psutil==5.9.8
pycodestyle==2.12.0
pydantic==2.7.4
pydantic_core==2.18.4
pydub==0.25.1
pyflakes==3.2.0
Pygments==2.18.0
PyJWT==2.8.0
pyparsing==3.1.2
python-dateutil==2.9.0.post0
python-dotenv==1.0.1
python-multipart==0.0.9
pytorch-lightning==2.3.0
```

```
pytz==2024.1
PyYAML==6.0.1
referencing==0.35.1
requests==2.32.3
requests-oauthlib==2.0.0
rfc3339-validator==0.1.4
rfc3986-validator==0.1.1
rich==13.7.1
rpds-py==0.18.1
ruff==0.4.9
s3transfer==0.10.1
semantic-version==2.10.0
sentry-sdk==2.5.1
setproctitle==1.3.3
setuptools==69.5.1
shellingham==1.5.4
simplejson==3.19.2
six==1.16.0
smmap==5.0.1
sniffio==1.3.1
starlette==0.37.2
swagger-spec-validator==3.0.3
sympy==1.12.1
tomlkit==0.12.0
toolz==0.12.1
torch==2.3.1
torchmetrics==1.4.0.post0
tqdm==4.66.4
triton==2.3.1
typer==0.12.3
types-python-dateutil==2.9.0.20240316
typing_extensions==4.12.2
tzdata==2024.1
ujson==5.10.0
uri-template==1.3.0
urllib3==2.2.2
uvicorn==0.30.1
uvloop==0.19.0
wandb==0.17.2
watchfiles==0.22.0
webcolors==24.6.0
```

```
websocket-client==1.8.0
websockets==11.0.3
wheel==0.43.0
yarl==1.9.4
```

## References

[1] Alinoori, Mahshid and Tzerpos, Vassilios. Music-STAR: A style translation system for audio-based re-instrumentation. In Rao, Preeti, Murthy, Hema A., Srinivasamurthy, Ajay, Bittner, Rachel M., Repetto, Rafael Caro, Goto, Masataka, Serra, Xavier, and Miron, Marius (eds.), *Proceedings of the 23rd International Society for Music Information Retrieval Conference, ISMIR 2022, Bengaluru, India, December 4-8, 2022*, pp. 419–426, 2022. URL https://archives.ismir.net/ismir2022/paper/000050.pdf.

[2] Azencot, Omri, Erichson, N. Benjamin, Lin, Vanessa, and Mahoney, Michael W. Forecasting sequential data using consistent Koopman autoencoders. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 475–485. PMLR, 2020. URL http://proceedings.mlr.press/v119/azencot20a.html.

[3] Bengio, Yoshua, Courville, Aaron C., and Vincent, Pascal. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, 2013. doi: 10.1109/TPAMI.2013.50. URL https://doi.org/10.1109/TPAMI.2013.50.

[4] Berman, Nimrod, Naiman, Ilan, and Azencot, Omri. Multifactor sequential disentanglement via structured Koopman autoencoders. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=6fuPIe9tbnC.

[5] Bhagat, Sarthak, Uppal, Shagun, Yin, Zhuyun, and Lim, Nengli. Disentangling multiple features in video sequences using Gaussian processes in variational autoencoders. In Vedaldi, Andrea, Bischof, Horst, Brox, Thomas, and Frahm, Jan-Michael (eds.), *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXIII*, volume 12368 of *Lecture Notes in Computer Science*, pp. 102–117. Springer, 2020. doi: 10.1007/978-3-030-58592-1_7. URL https://doi.org/10.1007/978-3-030-58592-1_7.

[6] Bonnici, Russell Sammut, Benning, Martin, and Saitis, Charalampos. Timbre transfer with variational auto encoding and cycle-consistent adversarial networks. In *International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, July 18-23, 2022*, pp. 1–8. IEEE, 2022. doi: 10.1109/IJCNN55064.2022.9892107. URL https://doi.org/10.1109/IJCNN55064.2022.9892107.

[7] Brunton, Steven L., Budisic, Marko, Kaiser, Eurika, and Kutz, J. Nathan. Modern Koopman theory for dynamical systems. *SIAM Rev.*, 64(2):229–340, 2022. doi: 10.1137/21M1401243. URL https://doi.org/10.1137/21m1401243.

[8] Burgess, Chris and Kim, Hyunjik. 3D Shapes dataset, 2018. URL https://github.com/google-deepmind/3d-shapes.

[9] Chen, Shuangshuang and Guo, Wei. Auto-encoders in deep learning—a review with new perspectives. *Mathematics (Basel)*, 11(8):1777, 2023. ISSN 2227-7390.

[10] Comanducci, Luca, Antonacci, Fabio, and Sarti, Augusto. Timbre transfer using image-to-image denoising diffusion implicit models. In Sarti, Augusto, Antonacci, Fabio, Sandler, Mark, Bestagini, Paolo, Dixon, Simon,

Liang, Beici, Richard, Gaël, and Pauwels, Johan (eds.), *Proceedings of the 24th International Society for Music Information Retrieval Conference, ISMIR 2023, Milan, Italy, November 5-9, 2023*, pp. 257–263, 2023. doi: 10.5281/ZENODO.10265271. URL `https://doi.org/10.5281/zenodo.10265271`.

[11] Dey, Sourya and Davis, Eric William. DLKoopman: A deep learning software package for Koopman theory. In Matni, Nikolai, Morari, Manfred, and Pappas, George J. (eds.), *Learning for Dynamics and Control Conference, L4DC 2023, 15-16 June 2023, Philadelphia, PA, USA*, volume 211 of *Proceedings of Machine Learning Research*, pp. 1467–1479. PMLR, 2023. URL `https://proceedings.mlr.press/v211/dey23a.html`.

[12] Engel, Jesse H., Hantrakul, Lamtharn, Gu, Chenjie, and Roberts, Adam. DDSP: Differentiable digital signal processing. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL `https://openreview.net/forum?id=B1x1ma4tDr`.

[13] Hsieh, Jun-Ting, Liu, Bingbin, Huang, De-An, Fei-Fei, Li, and Niebles, Juan Carlos. Learning to decompose and disentangle representations for video prediction. In Bengio, Samy, Wallach, Hanna M., Larochelle, Hugo, Grauman, Kristen, Cesa-Bianchi, Nicolò, and Garnett, Roman (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 515–524, 2018. URL `https://proceedings.neurips.cc/paper/2018/hash/496e05e1aea0a9c4655800e8a7b9ea28-Abstract.html`.

[14] Huang, Sicong, Li, Qiyang, Anil, Cem, Bao, Xuchan, Oore, Sageev, and Grosse, Roger B. TimbreTron: A WaveNet(CycleGAN(CQT(audio))) pipeline for musical timbre transfer. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=S1lvm305YQ`.

[15] Hung, Yun-Ning, Chen, Yi-An, and Yang, Yi-Hsuan. Learning disentangled representations for timbre and pitch in music audio. *CoRR*, abs/1811.03271, 2018. URL `http://arxiv.org/abs/1811.03271`.

[16] Hung, Yun-Ning, Chiang, I-Tung, Chen, Yi-An, and Yang, Yi-Hsuan. Musical composition style transfer via disentangled timbre representations. In Kraus, Sarit (ed.), *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 4697–4703. ijcai.org, 2019. doi: 10.24963/IJCAI.2019/652. URL `https://doi.org/10.24963/ijcai.2019/652`.

[17] Ichinaga, Sara M., Andreuzzi, Francesco, Demo, Nicola, Tezzele, Marco, Lapo, Karl, Rozza, Gianluigi, Brunton, Steven L., and Kutz, J. Nathan. PyDMD: A Python package for robust dynamic mode decomposition. *CoRR*, abs/2402.07463, 2024. doi: 10.48550/ARXIV.2402.07463. URL `https://doi.org/10.48550/arXiv.2402.07463`.

[18] Kingma, Diederik P. and Welling, Max. Auto-encoding variational Bayes. In Bengio, Yoshua and LeCun, Yann (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL `http://arxiv.org/abs/1312.6114`.

[19] Koopman, Bernard O. Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences - PNAS*, 17(5):315–318, 1931. ISSN 0027-8424.

[20] Lezama, José. Overcoming the disentanglement vs reconstruction trade-off via Jacobian supervision. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=Hkg4W2AcFm`.

[21] Li, Yingzhen and Mandt, Stephan. Disentangled sequential autoencoder. In Dy, Jennifer G. and Krause, Andreas (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5656–5665. PMLR, 2018. URL `http://proceedings.mlr.press/v80/yingzhen18a.html`.

[22] Lian, Jiachen, Zhang, Chunlei, and Yu, Dong. Robust disentangled variational speech representation learning for zero-shot voice conversion. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*, pp. 6572–6576. IEEE, 2022. doi: 10.1109/ICASSP43922.2022.9747272. URL `https://doi.org/10.1109/ICASSP43922.2022.9747272`.

[23] Liu, Yong, Li, Chenyu, Wang, Jianmin, and Long, Mingsheng. Koopa: Learning non-stationary time series dynamics with Koopman predictors. In Oh, Alice, Naumann, Tristan, Globerson, Amir, Saenko, Kate, Hardt, Moritz, and Levine, Sergey (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL `http://papers.nips.cc/paper_files/paper/2023/hash/28b3dc0970fa4624a63278a4268de997-Abstract-Conference.html`.

[24] Luo, Yin-Jyun, Agres, Kat, and Herremans, Dorien. Learning disentangled representations of timbre and pitch for musical instrument sounds using Gaussian mixture variational autoencoders. In Flexer, Arthur, Peeters, Geoffroy, Urbano, Julián, and Volk, Anja (eds.), *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands, November 4-8, 2019*, pp. 746–753, 2019. URL `http://archives.ismir.net/ismir2019/paper/000091.pdf`.

[25] Luo, Yin-Jyun, Cheuk, Kin Wai, Nakano, Tomoyasu, Goto, Masataka, and Herremans, Dorien. Unsupervised disentanglement of pitch and timbre for isolated musical instrument sounds. In Cumming, Julie, Lee, Jin Ha, McFee, Brian, Schedl, Markus, Devaney, Johanna, McKay, Cory, Zangerle, Eva, and de Reuse, Timothy (eds.), *Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020, Montreal, Canada, October 11-16, 2020*, pp. 700–707, 2020. URL `http://archives.ismir.net/ismir2020/paper/000162.pdf`.

[26] Luo, Yin-Jyun, Hsu, Chin-Cheng, Agres, Kat, and Herremans, Dorien. Singing voice conversion with disentangled representations of singer and vocal technique using variational autoencoders. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pp. 3277–3281. IEEE, 2020. doi: 10.1109/ICASSP40776.2020.9054582. URL `https://doi.org/10.1109/ICASSP40776.2020.9054582`.

[27] Luo, Yin-Jyun, Ewert, Sebastian, and Dixon, Simon. Towards robust unsupervised disentanglement of sequential data - A case study using music audio. In Raedt, Luc De (ed.), *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pp. 3299–3305. ijcai.org, 2022. doi: 10.24963/IJCAI.2022/458. URL `https://doi.org/10.24963/ijcai.2022/458`.

[28] Luo, Yin-Jyun, Ewert, Sebastian, and Dixon, Simon. Unsupervised pitch-timbre disentanglement of musical instruments using a Jacobian disentangled sequential autoencoder. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2024, Seoul, Republic of Korea, April 14-19, 2024*, pp. 1036–1040. IEEE, 2024. doi: 10.1109/ICASSP48485.2024.10447564. URL `https://doi.org/10.1109/ICASSP48485.2024.10447564`.

[29] Matthey, Loic, Higgins, Irina, Hassabis, Demis, and Lerchner, Alexander. dSprites: Disentanglement testing sprites dataset, 2017. URL https://github.com/google-deepmind/dsprites-dataset.

[30] Mor, Noam, Wolf, Lior, Polyak, Adam, and Taigman, Yaniv. A universal music translation network. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=HJGkisCcKm.

[31] Naiman, Ilan, Erichson, N. Benjamin, Ren, Pu, Mahoney, Michael W., and Azencot, Omri. Generative modeling of regular and irregular time series data via Koopman VAEs. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=eY7sLb0dVF.

[32] Pan, Shaowu, Kaiser, Eurika, de Silva, Brian M., Kutz, J. Nathan, and Brunton, Steven L. PyKoopman: A Python package for data-driven approximation of the Koopman operator. *J. Open Source Softw.*, 9(96):5881, 2024. doi: 10.21105/JOSS.05881. URL https://doi.org/10.21105/joss.05881.

[33] Pati, Ashis, Gururani, Siddharth Kumar, and Lerch, Alexander. dMelodies: A music dataset for disentanglement learning. In Cumming, Julie, Lee, Jin Ha, McFee, Brian, Schedl, Markus, Devaney, Johanna, McKay, Cory, Zangerle, Eva, and de Reuse, Timothy (eds.), *Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020, Montreal, Canada, October 11-16, 2020*, pp. 125–133, 2020. URL http://archives.ismir.net/ismir2020/paper/000300.pdf.

[34] Peeters, Geoffroy, Giordano, Bruno Lucio, Susini, Patrick, Misdariis, Nicolas, and McAdams, Stephen. The Timbre Toolbox: Extracting audio descriptors from musical signals. *The Journal of the Acoustical Society of America*, 130(5):2902–2916, 2011. ISSN 0001-4966.

[35] Schmid, Peter J. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656(August):5–28, 2010. ISSN 0022-1120.

[36] Schmid, Peter J. and Sesterhenn, Joern L. Dynamic mode decomposition of numerical and experimental data. In *61st Annual Meeting of the APS Division of Fluid Dynamics*, volume 53 of *Bulletin of the American Physical Society*, pp. 208, San Antonio, Texas, United States of America, 11 2008. American Physical Society.

[37] Semmelrock, Harald, Kopeinik, Simone, Theiler, Dieter, Ross-Hellauer, Tony, and Kowald, Dominik. Reproducibility in machine learning-driven research. *CoRR*, abs/2307.10320, 2023. doi: 10.48550/ARXIV.2307.10320. URL https://doi.org/10.48550/arXiv.2307.10320.

[38] Semmelrock, Harald, Ross-Hellauer, Tony, Kopeinik, Simone, Theiler, Dieter, Haberl, Armin, Thalmann, Stefan, and Kowald, Dominik. Reproducibility in machine learning-based research: Overview, barriers and drivers. *CoRR*, abs/2406.14325, 2024. doi: 10.48550/ARXIV.2406.14325. URL https://doi.org/10.48550/arXiv.2406.14325.

[39] Simon, Mathieu Cyrille, Frossard, Pascal, and Vleeschouwer, Christophe De. Sequential representation learning via static-dynamic conditional disentanglement. *CoRR*, abs/2408.05599, 2024. doi: 10.48550/ARXIV.2408.05599. URL https://doi.org/10.48550/arXiv.2408.05599.

[40] Sinha, Koustuv, Bleeker, Maurits, Bhargav, Samarth, Forde, Jessica Zosa, Raparthy, Sharath Chandra, Dodge, Jesse, Pineau, Joelle, and Stojnic, Robert. ML reproducibility challenge 2022. 7 2023. doi: 10.5281/ZENODO.8200058. URL https://zenodo.org/record/8200058.

[41] Szabó, Attila, Hu, Qiyang, Portenier, Tiziano, Zwicker, Matthias, and Favaro, Paolo. Understanding degeneracies and ambiguities in attribute transfer. In Ferrari, Vittorio, Hebert, Martial, Sminchisescu, Cristian, and Weiss, Yair (eds.), *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part V*, volume 11209 of *Lecture Notes in Computer Science*, pp. 721–736. Springer, 2018. doi: 10.1007/978-3-030-01228-1_43. URL `https://doi.org/10.1007/978-3-030-01228-1_43`.

[42] Tanaka, Keitaro, Nishikimi, Ryo, Bando, Yoshiaki, Yoshii, Kazuyoshi, and Morishima, Shigeo. Pitch-timbre disentanglement of musical instrument sounds based on VAE-based metric learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pp. 111–115. IEEE, 2021. doi: 10.1109/ICASSP39728.2021.9414059. URL `https://doi.org/10.1109/ICASSP39728.2021.9414059`.

[43] Tanaka, Keitaro, Bando, Yoshiaki, Yoshii, Kazuyoshi, and Morishima, Shigeo. Unsupervised disentanglement of timbral, pitch, and variation features from musical instrument sounds with random perturbation. In *2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 709–716. Asia-Pacific of Signal and Information Processing Association (APSIPA), 2022. ISBN 6165904777.

[44] Tayal, Kshitij, Renganathan, Arvind, Ghosh, Rahul, Jia, Xiaowei, and Kumar, Vipin. Koopman invertible autoencoder: Leveraging forward and backward dynamics for temporal modeling. In Chen, Guihai, Khan, Latifur, Gao, Xiaofeng, Qiu, Meikang, Pedrycz, Witold, and Wu, Xindong (eds.), *IEEE International Conference on Data Mining, ICDM 2023, Shanghai, China, December 1-4, 2023*, pp. 588–597. IEEE, 2023. doi: 10.1109/ICDM58522.2023.00068. URL `https://doi.org/10.1109/ICDM58522.2023.00068`.

[45] Tonekaboni, Sana, Li, Chun-Liang, Arik, Sercan Ö., Goldenberg, Anna, and Pfister, Tomas. Decoupling local and global representations of time series. In Camps-Valls, Gustau, Ruiz, Francisco J. R., and Valera, Isabel (eds.), *International Conference on Artificial Intelligence and Statistics, AISTATS 2022, 28-30 March 2022, Virtual Event*, volume 151 of *Proceedings of Machine Learning Research*, pp. 8700–8714. PMLR, 2022. URL `https://proceedings.mlr.press/v151/tonekaboni22a.html`.

[46] Tu, Jonathan H., Rowley, Clarence W., Luchtenburg, Dirk M., Brunton, Steven L., and Kutz, J. Nathan. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014. ISSN 2158-2505. doi: 10.3934/jcd.2014.1.391. URL `http://dx.doi.org/10.3934/jcd.2014.1.391`.

[47] Wang, Rui, Dong, Yihe, Arik, Sercan Ö., and Yu, Rose. Koopman neural operator forecaster for time-series with temporal distributional shifts. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL `https://openreview.net/forum?id=kUmdmHxK5N`.

[48] Woo, Gerald, Liu, Chenghao, Sahoo, Doyen, Kumar, Akshat, and Hoi, Steven C. H. CoST: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL `https://openreview.net/forum?id=PilZY3omXV2`.

[49] Yamada, Masanori, Kim, Heecheol, Miyoshi, Kosuke, Iwata, Tomoharu, and Yamakawa, Hiroshi. Disentangled representations for sequence data using information bottleneck principle. In Pan, Sinno Jialin and Sugiyama,

Masashi (eds.), *Proceedings of The 12th Asian Conference on Machine Learning, ACML 2020, 18-20 November 2020, Bangkok, Thailand*, volume 129 of *Proceedings of Machine Learning Research*, pp. 305–320. PMLR, 2020. URL `http://proceedings.mlr.press/v129/yamada20a.html`.

# תוכן עניינים

# שיפור התרה רבת-מרכיבים סדרתית מבוססת קופמן עם אופן תנודה סטטי יחיד וזיקוק המרחב הסמוי

**עמוס חביב חסון**

חיבור לשם קבלת התואר

מוסמך למדעים

באוניברסיטת בן-גוריון בנגב

2024

## תקציר

למידת ייצוגים משמשת לגילוי אוטומטי של מאפיינים שימושיים מתוך נתונים גולמיים לצורך ייעול פיתרון משימות כגון סיווג וחיזוי. ייצוגים מותרים, בהם מאפיינים מובחנים קשורים למרכיבים בלתי-תלויים של השונות בנתונים, מציעים גישה מבטיחה לשיפור ההכללה של מודלים, במיוחד בנתונים סדרתיים. אולם למידה של ייצוגים כאלו באופן בלתי-מונחה היא מאתגרת, במיוחד בשל המחסור בנתונים מתוייגים. בעבודה זו אנו מתמקדים במודל התרת קופמן מובנית (הק״מ), שמממש פירוק אופני תנודה דינמי (פאת״ד) באמצעות תאוריית קופמן כחלק ממקודד עצמי על מנת להשיג התרה סדרתית בלתי-מונחית. אנו מזהים בעיות מפתח במימוש המקורי של הק״מ ומציעים מימוש מחדש שפותר אותן. יתרה מכך, אנו מציגים מודל התרת קופמן מובנית עם אופן תנודה סטטי יחיד (הק״מ-אתס״י), שמשלב התאמות לשלבי הפאת״ד וחילוץ המרחב הסמוי. לצורך שיפור חקירת המרחב הסמוי, אנו מציעים בנוסף אלגוריתם חמדן לחקירת המרחב הסמוי. המודל שלנו נאמד מול ארבעה אוספי נתונים של התרה רבת-מרכיבים סדרתית, ומפגין שיפור במועילות ביחס למודל הק״מ המקורי. בנוסף לכך, אנו תורמים לפיתרון אתגר ההדירות בלמידה חישובית על-ידי הצעת תקן לדיווח מקיף של סביבות ניסוייות.

אוניברסיטת בן-גוריון בנגב
הפקולטה למדעי הטבע
המחלקה למדעי המחשב

# שיפור התרה רבת-מרכיבים סדרתית מבוססת קופמן עם אופן תנודה סטטי יחיד וזיקוק המרחב הסמוי

חיבור לשם קבלת התואר

מוסמך למדעים

עמוס חביב חסון

בהנחיית :

מרצה-בכיר ד״ר עומרי אזנקוט

דצמבר 2024

כסלו ה׳תשפ״ה